UDC 519.25

Improving the Energy Efficiency of the Smart Buildings with the Boosting Algorithms

Eugeny Yu. Shchetinin*, Vladimir S. Melezhik†‡, Leonid A. Sevastyanov‡†

* *Department of Data Analysis, Decisions and Finacial Technologies*
Financial University under the Gouvernment of the Russian Federation
Leningradsky pr. 49, Moscow, 111123, Russia
† *Bogolyubov Laboratory of Theoretical Physics*
Joint Research Institute for Nuclear Research
Joliot-Curie 6, Dubna, 141980, Moscow region, Russia
‡ *Institute of Applied Mathematics and Communications Technology*
Peoples' Friendship University of Russia (RUDN University)
Miklukho-Maklaya str. 6, Moscow, 117198, Russia

Email: riviera-molto@mail.ru, melezhik@theor.jinr.ru, sevastianov-la@rudn.ru

In the promotion of modern real estates with a high level of energy consumption are the most important assessment of their energy efficiency. Increasing the level of technology in commercial buildings with digital infrastructure of accounting, control and management energy consumption has led to increased availability of data produced by the digital sensors. All this opens up huge opportunities for using of advanced mathematical models and machine learning methods that would improve the accuracy of forecasts of electricity consumption by commercial buildings, and thus improve estimates of energy saving. One of the most powerful algorithms in machine learning is gradient boosting (GBM). In this paper on GBM basis a method of the energy consumption profile modeling is proposed both for a separate building and for business centers. To evaluate its effectiveness advanced computer experiments were performed on real data of the energy consumption of commercial buildings. For this purpose different periods of model training were used, and its prediction accuracy was analyzed by several criteria. The results showed that the use of our model improved the accuracy forecasts of energy savings in more than 80 percent of cases compared to regression and random forest models.

Key words and phrases: energy consumption, smart building, smart meters, gradient boosting, random forests.

In: K. E. Samouylov, L. A. Sevastianov, D. S. Kulyabov (eds.): Selected Papers of the 12th International Workshop on Applied Problems in Theory of Probabilities and Mathematical Statistics (Summer Session) in the framework of the Conference "Information and Telecommunication Technologies and Mathematical Modeling of High-Tech Systems", Lisbon, Portugal, October 22–27, 2018, published at http://ceur-ws.org

1. Introduction

One of the most important directions of economic development is to improve the energy efficiency of the industrial and consumer sectors of the economy. The state program of the Russian Federation "Energy saving and increase of energy efficiency for the period up to 2030" was approved as one of the most important directions of development of the Russian economy. Several energy efficiency programmes have been implemented to reduce the environmental impact and costs of the commercial building sector. For example, long-term energy saving targets have been set at the state and Federal levels in Russia, and these targets should be achieved with the help of energy efficiency programs. Measuring and verifying energy efficiency is a process of assessing sav-ings and is therefore crucial to determining the value of energy efficiency for building owners, utility tariff payers and service providers. Today, the growing availability of data from smart meters and devices, combined with data mining, allows the process to be optimized by increasing the level of automation while maintaining or improving the accuracy of the forecasting consumption [1–3].

The basic models used in estimating the energy consumption profiles, are regression models that link the consumption of energy in buildings with such parameters as, for example, the temperature of the external environment, humidity, characteristics of the building, etc. [4, 5]. Traditionally, for building such models, we used data of monthly bills for utilities, however, the increase in data availability from smart meters with an hour and 15-minute intervals helped us to create new methods for more accurate forecasts. In recent years, several approaches to base energy modeling using smart meter data have appeared in the literature. These methods are based on traditional linear regression, nonlinear regression, and machine learning methods. The linear regression model described in includes the time of day, the day of the week, and two temperature functions that allow different combinations of heating and cooling. It can also include humidity and holidays as variables. This model is well described by the usual regression, which is a common practice in such cases [6], see also [7, 8].

Over the past two decades, significant progress has been made in the development of new methods of machine learning, among which the most promising in terms of prediction accuracy are the approaches of the family of ensemble learning algorithms. Ensemble methods construct a model by training several relatively simple models, and then combine them to create a more complex model with higher predictive properties. The well-known ensemble learning algorithms use bagging [10], random forests [11], randomized trees and boosting [12]. Bagging, extreme trees and random forests are based on a simple averaging of the base student, while enhancement algorithms are built on a constructive iterative strategy. Although these ensemble machine learning algorithms have been used with great success in many fields, they are just starting to be applied to problems in the field of energy building modeling. For example, in [2] the authors used a random forest to predict the hourly energy consumption, the random forest algorithm was applied to detect anomalies in the energy consumption of the building [13]. In this paper an attempt is made to create a new numerical algorithm for selecting and fine-tuning the parameters of the model of the basic energy consumption profile of a conglomerate of commercial buildings. To solve this problem, we used the gradient boosting algorithm and developed a number of algorithms for testing the accuracy of predicting energy consumption in buildings.

2. Advanced graident boosting models

Efficient concept of decision trees, also known as regression trees, consist of partitioning the input parameters space into distinct and non-overlapping regions following a set of if-then rules. The splitting rules identify regions that have the most homogeneous response to the predictor, and within each region a simple model, such as a constant, is fitted. The split points are chosen to minimize a loss-function, which in the case of regression trees is usually the mean squared error. The splitting continues until a stopping criterion is reached, e.g., the number of training points within a region

reaches some defined threshold. These different splitting steps correspond to the depth of the tree. To make a prediction for new data points, the data are split following the trained split points, and the same constants in the terminal nodes are used to make the predictions.

The use of decision trees as a regression technique has several advantages, one of which is that the splitting rules represent an intuitive and very interpretable way to visualize the results. In addition, by their design, they can handle simultaneously numerical and categorical input parameters. They are robust to outliers and can efficiently deal with missing data in the input parameters space. The decision tree's hierarchical structure automatically models the interaction between the input parameters and naturally performs variable selection, e.g., if an input parameter is never used during the splitting procedure, then the prediction does not depend on this input parameter. Finally, decision trees algorithms are simple to implement and computationally efficient with a large amount of data.

The boosting algorithm was first proposed in [3] for classification problems. Its basic principle is that several simple models, called "weak learning models", to be merged into one iterative scheme for the selection of parameters with the aim of obtaining the so-called "strong learning model", i.e. models with better prediction accuracy. Thus, the GBM algorithm iteratively adds a new decision tree (i.e. "weak learner") at each step, which best reduces the loss function. Specifically, in a regression model, the algorithm starts with model initialization, which is typically a decision tree minimizing the loss function (RMSE), and then at each step, a new decision tree is adjusted to the current residual and added to the previous model to update the residuals. The algorithm continues to run until the maximum number of iterations is reached or the specified precision is reached. It means that at each new step, the decision trees added to the model in the previous steps are fixed. Thus, the model can be improved in those parts of it where it still does not assess the residuals.

The GBM algorithm will be more efficient if at each iteration the contribution of the added decision tree is taken into account using some hyperparameter (shrinkage rate) that can intuitively characterize the learning model rate. The idea of the hyperparameter selection procedure is that more small steps provide higher accuracy than fewer large steps. The parameter can be from 0 to 1, and the smaller it is, the more accurate the model will be. However, choosing a stronger shrinkage implies more iterations to achieve the required accuracy, since the value is inversely proportional to the number of iterations. Another way to improve the prediction accuracy of the GBM algorithm is to add randomization to the estimation process. At each iteration, instead of using the full data set, a subsample (without replacement as usual) is applied to evaluate the decision tree. However, to assess the impact of reducing the number of data points on the quality of the fit of the model, it is necessary to check several subsamples of different dimensions. GB model have four hyperparameters that must be tuned: (1) d - the depth of the decision trees, which also determines the maximum order of the model; (2) K-the number of iterations, which also corresponds to the number of decision trees; (3) the learning rate, which is usually a small positive value between 0 and 1, the reduction of which leads to a slowdown in the estimation procedure, thus requiring the user to increase K; (4) the piece of data that is used in each iteration step. The following section describes the purpose of configuring these hyperparameters and the method used for it as follows.

3. GBM hyperparameters tuning

As with any forecasting method, the overfitting problem is relevant for the GBM algorithm. Overfitting is usually a disadvantage of an overly complex model. In the case of the GBM model, this can happen if too many iterations K and too deep decision trees d are chosen [4]. So, the main purpose of this paper to choose the right combination of hyperparameters to avoid overfitting and at the same time provide the best predictive accuracy for our models. Several approaches have been introduced and studied in

the literature on statistics and machine learning. However, the most popular and conceptually easy to understand method is the a grid search. This approach consists of defining a grid of combinations of hyperparameter values, building a model for each combination, and selecting the optimal combination using metrics that quantify the model in terms of prediction accuracy. Dataset should be divided into two samples: a training sample and a test sample. In practice, however, it is rarely possible to allocate enough data points to accurately assess the predictive performance of models without affecting the quality of the estimate. When the number of observations is not enough, the decreasing the size of the training set can result in poor estimates. Cross-validation (CV), especially k-fold CV, is the most effective method to solve this problem. The k-fold cross-validation method involves randomly dividing a data set into k subsamples of approximately the same size, called folds. The first model is evaluated using k-1 folds as the training dataset, and the left part (test sample) is used to determine the accuracy of the prediction. In this paper we use as the accuracy metric the root mean square deviation (RMSE) in the following form $\sqrt{\frac{1}{n}\sum_{i=1}^{n}(y_i-\widetilde{y})^2}$, where $\widetilde{y}$ is predicted value, y_i - real data set value. This procedure is repeated k times, and each time a different part is used as a test sample. The k-fold cross-validation method determines the standard deviation by the equation $RMSE\,(CV)=\frac{1}{k}\sum_{i=1}^{k}(RMSE_i)$. Simplified illustration of this algorithm is provided by the following pseudocode:

1. The user states the depth of the decision trees d, the number of iterations K, the learning rate ν and the size of the subsample η;

2. Initial step: equal $r_0=y$ and $\widehat{f}_0=0$. The mean value of y was proposed as the initial value of $\widehat{f}$;

3. For $k=1,2,...K$ do the following:

a) Randomly select a subsample (x_i,y_i), $i=1,...,N$. from the full set of training data, where N is the number of data points corresponding to the fraction η;

b) Using (x_i,y_i) fit the decision tree $\widehat{f}^k$ of depth d to the residuals z^k;

c) Update $\widehat{f}$ by adding a decision tree to the model $\widehat{f}\,(x)\leftarrow\widehat{f}\,(x)+\alpha\widehat{f}^k\,(x)$;

d) Update remainder $z^k=z^{k-1}-\alpha\widehat{f}^k\,(x)$;

4. Stop.

4. Applications for commercial buildings

For each building, time series data ware split into training and forecasting periods. The forecast period was defined as the last 12 months of available data. Models are trained using two different training periods, which are 3, 6 and 12 months. All buildings in the database have 36 months of electricity consumption and outdoor temperature data. The GBM hyperparameters were configured automatically using the grid search method with cross-validation methods (CV). Thus, the depth of decision trees d was chosen from the set $(3,..,10)$, the learning rate ν was chosen between 0.05 and 1, the number of iterations K was chosen between 10 iterations and 1000 with a step of 10 iterations. Three variants of standart k-fold–block CV were used: 5-fold-block CV, 5-fold-CV with GBM-day as block for CV and 5-fold-block with GBM-week as a block for CV. The results show that the overall value of 0.1 for the learning rate was too small because the algorithm was too sensitive for both the number of iterations and the depth of the decision trees. It also seems that at a learning rate of 0.2 and a decision tree depth of 5 (optimal depth), the algorithm did not reach the optimal number of iterations at K=500. This probably explains why the optimal learning rate for this example was 0.5, which has a higher convergence rate together with smaller computing time. In addition, the decrease in prediction accuracy is due to the fact that at some point, by increasing the number of iterations and the complexity of the model, the algorithm begins to adjust the training data too much, except for the learning rate $\nu=1$ with the depth of the decision tree 5. These results are demostrated on Figure 1.

The RF model was developed using the randomForest R package [15]. As input variables for RF were considered outdoor air temperature, time of week and some dummy variables. The two high-frequency hyperparameters considered during the setup process were: the number of input variables randomly selected as candidates on each split and the number of trees to grow (ntree). To configure these two hyperparameters, the search grid method and the block K-fold cross-validation method were used with the day defined as a block and with k = 5. Thus, mtry is selected in the set 1,2,3, and ntree is selected in the set 50, 250, 500. Please note that instead of the 5-fold CV method CV was used with a 5x block; this choice is motivated by the fact that the empirical results showed that the use of 5-time units has improved the accuracy of the RF models. These results are demostrated on Figure 2. Some computer program code realising base algorithms used in this paper is presented in section 6.

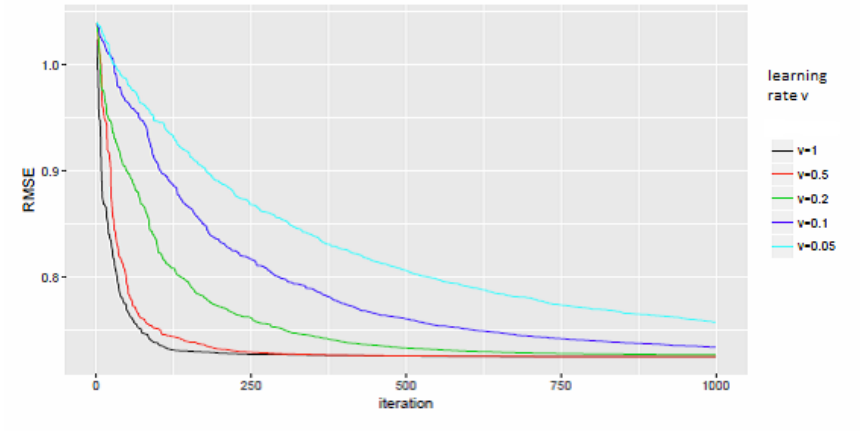

Figure 1. Accuracy prediction RMSE dependence on the number of iterations K for different learning rates $\nu = (0.05, 0.1, 02, 0.5, 1)$.

Accuracy rates R^2 , CV (RMSE) were calculated for the entire data set and demonstrated accuracy reduction as the training period was reduced from 12 months to 6 months. Our computer experiments have shown that R^2 of the GBM models superior to the RF and regression models. This is especially remarkable for the GBM-1d and GBM-7d models, and as expected, the GBM model using the standard k-fold CV has less accuracy than the other two versions that use the k-fold block CV. With the training period reduced to 6 months, a significant decrevarase was observed in the standard GBM model (with standard K-fold CV) and a slight decrease in R^2 for GBM-1d, GBM7d, and RF models, while accuracy improved in the regression model, which means that for this dataset, the regression model does not improve accuracy with an increase in the number of observations. With regard to the evaluation findings on the CV(RMSE), according to our calculations, models GBM-1 and GBM-7d are slightly superior to the models RF and GBM. The accuracy of the GBM models improved significantly when their training period was increased from 6 months to 12 months, while the accuracy of the regression model was slightly reduced. In the Table 1 are shown the number of buildings as a percentage for which GBM models seemed more accurate than regression and RF models. Recall that higher values are desirable for R^2, whereas for CV (RMSE) values it is desirable to have them close to zero. For RMSE(CV), the columns represent the

percentage of buildings with a lower RMSE(CV) than the regression model. These results confirm that GBM-1d and GBM-7d algorithms have better accuracy than regression and RF models.

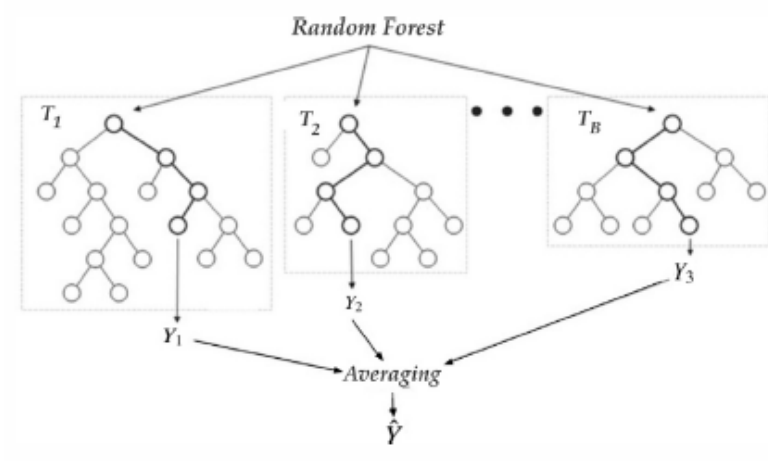

Figure 2. Results of Random Forest model applications $k = 3$.

5. Main results and conclusions

A method of constructing a model of the basic profile of electricity consumption by large commercial centers and business buildings is proposed. It is based on a gradient boosting algorithm with adaptive hyperparameters tuning procedure using the K-fold-blocks cross-validation. The efficiency and effectiveness of GB in solving the problem of energy efficiency has been tested on both model and real data. The GB model showed higher prediction accuracy than regression and random forest models for all tested training periods. The results of computer experiments have shown that the use of the GBM model can improve the accuracy of energy efficiency assessment of the entire building. See Table 1 for details. Our great finding is that the use of a 6-month training period to build GBM models resulted in a slight decrease in the accuracy of the energy consumption forecast, compared to those obtained during the 12-month training period, which is usually used for the entire building. This allows to reduce the total time required for the evaluation of energy savings of the entire complex of buildings.

Comparison of different algorithms of hyperparameters tuning showed that it is important to take into account autocorrelation of energy consumption time series. Indeed, the results suggest that the use of standard K-fold CV cross-validation reduces the accuracy of the GBM algorithm. This is due to the fact that when using the standard K-fold approach cross validation observations in test and training data sets are not independent (due to autocorrelation of measurements obtained from smart sensors), which leads to overfitting of the model. It has also been shown that the difference in the use of the forecast block as a week or day does not have a significant impact on the accuracy. Therefore, we can conclude that in most cases using the day as the default block size is a good choice. It is known that one of the main advantages of the ensemble tree model is its flexibility and reliability when used with a large number of

Table 1

Models accuracy forecast for 6 an 12 month periods.

Model/Criteria	R^2 $6m$	$RMSE(CV)$ $6m$	R^2 $12m$	$RMSE(CV)$ $12m$
GBM	33	47	61	76
$GBM - 1day$	57	63	67	81
$GBM - 7day$	77	70	81	86
$R - Forest$	28	40	35	48
$Regression$	17	30	27	38

input parameters. Also, compared to models such as regression models, there is no need to modify the algorithm to handle additional input parameters such as building occupancy, humidity, or solar radiation. Instead, it is sufficient to include these variables in the input table of the algorithm without having to define a specific form of the model for each of the parameters, as is the case with most standard regression algorithms used in modern practical application. In addition, the ability to select GBM model variables allows you to include parameters that do not affect the model, without reducing the predictability of the model. Finally, the GB model has a number of obvious advantages over regression models in its ability to maintain accuracy for shorter training periods, improve overall accuracy with respect to energy efficiency indicators, and make it easy to include additional explanatory variables. Key areas of future work will be the application of the GBM model to address energy efficiency issues such as forecasting energy consumption, singular anomaly detection, and quantifying load reduction.

6. Program Code

PYTHON code for parameter estimation using grid search with block K-fold cross-validation

```
from __future__ import print_function

from sklearn import datasets
from sklearn.model_selection import train_test_split
from sklearn.model_selection import GridSearchCV
from sklearn.metrics import classification_report
from sklearn.svm import SVC

# Loading the elec-consumer dataset
digits = datasets.load_elec-consumer()

# To apply an classifier on this data, we need to flatten the image, to
# turn the data in a (samples, feature) matrix:
n_samples = len(elec-consumer.images)
X = elec-consumer.images.reshape((n_samples, -1))
y = elec-consumer.target

# Split the dataset in two equal parts
X_train, X_test, y_train, y_test = train_test_split(
    X, y, test_size=0.5, random_state=0)

# Set the parameters by cross-validation
tuned_parameters = [{'kernel': ['rbf'], 'gamma': [1e-3, 1e-4],
                     'C': [1, 10, 100, 1000]},
```

```
                    {'kernel': ['linear'], 'C': [1, 10, 100, 1000]}]

scores = ['precision', 'recall']

for score in scores:
    print("# Tuning hyper-parameters for  score)
    print()

    clf = GridSearchCV(SVC(), tuned_parameters, cv=5,
                       scoring='s_macro' score)
    clf.fit(X_train, y_train)

    print("Best parameters set found on development set:")
    print()
    print(clf.best_params_)
    print()
    print("Grid scores on development set:")
    print()
    means = clf.cv_results_['mean_test_score']
    stds = clf.cv_results_['std_test_score']
    for mean, std, params in zip(means, stds, clf.cv_results_['params']):
        print("0.3f (+/-0.03f) for r" (mean, std * 2, params))
    print()

    print("Detailed classification report:")
    print()
    print("The model is trained on the full development set.")
    print("The scores are computed on the full evaluation set.")
    print()
    y_true, y_pred = y_test, clf.predict(X_test)
    print(classification_report(y_true, y_pred))
    print()
```

PYTHON code for Gradient Boosting Machine Algorithm

```
import numpy as np
import matplotlib.pyplot as plt

from sklearn import ensemble
from sklearn import datasets

X, y = datasets.make_hastie_10_2(n_samples=12000, random_state=1)
X = X.astype(np.float32)

# map labels from {-1, 1} to {0, 1}
labels, y = np.unique(y, return_inverse=True)

X_train, X_test = X[:2000], X[2000:]
y_train, y_test = y[:2000], y[2000:]

original_params = {'n_estimators': 1000, 'max_leaf_nodes': 4,
 'max_depth': None, 'random_state': 2,'min_samples_split': 5}

plt.figure()

for label, color, setting in [('No shrinkage', 'orange',
     {'learning_rate': 1.0, 'subsample': 1.0}),
```

```
        ('learning_rate=0.1', 'turquoise',
        {'learning_rate': 0.1, 'subsample': 1.0}),
         ('subsample=0.5', 'blue',
         {'learning_rate': 1.0, 'subsample': 0.5}),
         ('learning_rate=0.1, subsample=0.5', 'gray',
         {'learning_rate': 0.1, 'subsample': 0.5}),
         ('learning_rate=0.1, max_features=2', 'magenta',
         {'learning_rate': 0.1, 'max_features': 2})]:
    params = dict(original_params)
    params.update(setting)

    clf = ensemble.GradientBoostingClassifier(**params)
    clf.fit(X_train, y_train)

    # compute test set deviance
    test_deviance = np.zeros((params['n_estimators'],), dtype=np.float64)

    for i, y_pred in enumerate(clf.staged_decision_function(X_test)):
        # clf.loss_ assumes that y_test[i] in {0, 1}
        test_deviance[i] = clf.loss_(y_test, y_pred)

    plt.plot((np.arange(test_deviance.shape[0]) + 1)[::5],
    test_deviance[::5], '-', color=color, label=label)

plt.legend(loc='upper left')
plt.xlabel('Boosting Iterations')
plt.ylabel('Test Set Deviance')
```

R code for GAM time series model presentation

```
library(mgcv)
gamst <- proc.time()
z <- as.vector(log(ru.ext$rate$total))
x <- 1:nrow(ru.ext$rate$total)
y <- 1:ncol(ru.ext$rate$total)
xy <- expand.grid(x, y)
ru.gam <- gam(z~s(xy[,1],xy[,2], bs='ts', k=12^2))
gamen <- proc.time()
gamel <- gamen['elapsed'] - gamst['elapsed']
cat("Gam time passed:", gamel, "\n")
persp(matrix(fitted(ru.gam), nrow=length(x), ncol=length(y)))
persp(matrix(residuals(ru.gam), nrow=length(x), ncol=length(y)))
levelplot(matrix(residuals(ru.gam), nrow=length(x), ncol=length(y)))
wireframe(
matrix(fitted(ru.gam), nrow=52, ncol=52),
xlab = expression(a),
ylab = expression(y),
zlab = expression(m),
screen = list(z = 20, x = -70, y = 3)
)
```

Acknowledgments

The publication has been prepared with the support of the "RUDN University Program 5-100".

References

1. C. W. Gelling, The Smart Grid: Enabling energy efficiency and demand response. The Fairmont Press Inc., 2009.
2. W.C. Hong,Intelligent Energy Demand Forecasting, Springer Verlag, London, 2013.
3. L. Breiman, Bagging predictors, Mach. Learn. 24 (2), 123–140, 1996.
4. J.H. Friedman, Stochastic gradient boosting, Comp. Stat. Data Anal. 38 (4), 367–378, 2002.
5. M. Kuhn, K. Johnson, Applied Predictive Modeling, Springer, New York, 2013.
6. Eu. Yu. Shchetinin, P.G. Lyubin, Fast two-dimensional smoothing with discrete cosine transform, Springer Communications in Computer and Information Science (CCIS), 678, 646-656. Springer, Berlin, 2016.
7. Gudkova, I., Samouylov, K., Buturlin, I., Borodakiy, V., Gerasimenko, M., Galinina, O., Andreev, S., Analyzing impacts of coexistence between M2M and H2H Communication on 3GPP LTE System, Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics), 8458, pp. 162-174, 2014
8. Naumov, V., Samouylov, K., Analysis of multi-resource loss system with state-dependent arrival and service rates, Probability in the Engineering and Informational Sciences, 31 (4), pp. 413-419, 2017.
9. P. Geurts, D. Ernst, L. Wehenkel, Extremely randomized trees, Mach. Learn. 63(1), 3–42, 2006.
10. A. Srivastav, A. Tewari, B. Dong, Baseline building energy modeling and localized uncertainty quantification using Gaussian mixture models, Energy Build. 65, 438–447, 2013.
11. H.X. Zhao, F. Magoulès, A review on the prediction of building energy consumption, Renew. Sustain. Energy Rev. 16(6),3586–3592, 2012.
12. Y. Heo, V.M. Zavala, Gaussian process modeling for measurement and verification of building energy savings, Energy Build. 53, 7–18, 2012.
13. S. Price, A. Mahone, N. Schlag, D. Suyeyasu, Time dependent valuation of energy for developing building efficiency standards, in: Report Prepared for the California Energy Commission, 2011.
14. Eu. Shchetinin, Cluster-based energy consumption forecasting in smart grids, Springer Communications in Computer and Information Science (CCIS), Springer, Berlin, 919, 46-656, 2018.
15. A. Liaw, M. Wiener, Classification and Regression by randomForest, R News, 2 (3),18–22, 2002.
16. R.E. Schapire, The strength of weak learnability, Mach. Learn. 5 (2), 197–227,1990.